



머신러닝 & 딥러닝 1

AI 학술동아리 <MLP>

- Index

- 1. 지도 학습 / 비지도 학습 / 강화 학습**
- 2. 훈련 세트, 테스트 세트 / 샘플링 편향**
- 3. 데이터 전처리**

1-1. Supervised Learning(지도 학습)

				
training data(훈련 데이터)	input(입력)	길이 feature(특성) 길이 :	10cm	20cm
		무게 feature(특성) 무게 :	300g	200g
	target(정답) /label	사과 :	0	바나나 : 1

여러 사과(바나나) 데이터 중에 하나의 사과(바나나) 데이터 - **sample(샘플)**

1-2. Unsupervised Learning(비지도 학습)

training data(훈련 데이터) — Input(입력)

길이 feature(특성)

길이 :

10cm

20cm

무게 :

300g

200g

무게 feature(특성)



정답값이 없음 / 무엇을 예측하는 것이 아니라 input data에서
어떤 특징을 찾는 데 주로 활용



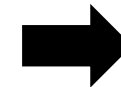
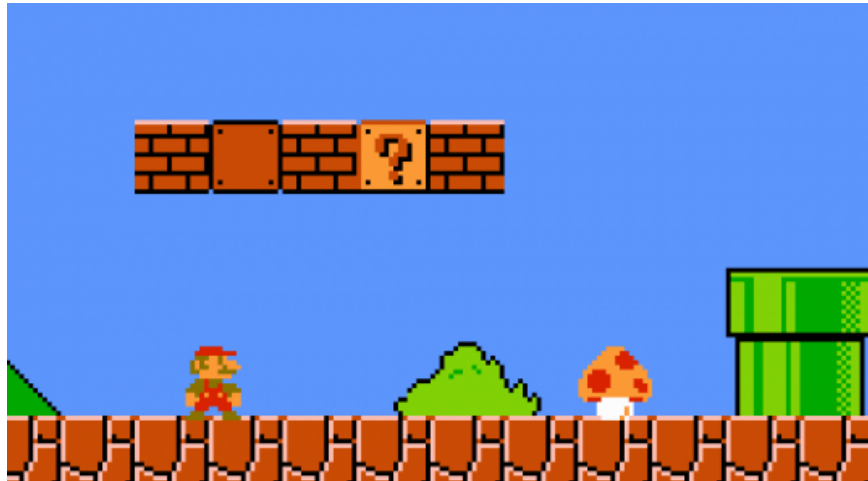
애는 왼쪽 애들이랑 비슷하네!

1-3. Reinforcement Learning(강화 학습)

간단하게 **강화학습**은 **trial and error(시행착오)**를
겪으면서 점점 성능이 좋은 쪽으로 학습해가는 것을 말함

사람이 같은 게임을 반복적으로 연습함으로써
점점 그 게임을 클리어할 수 있게 되고,
클리어하는 시간이 단축되는 것이라 비슷함

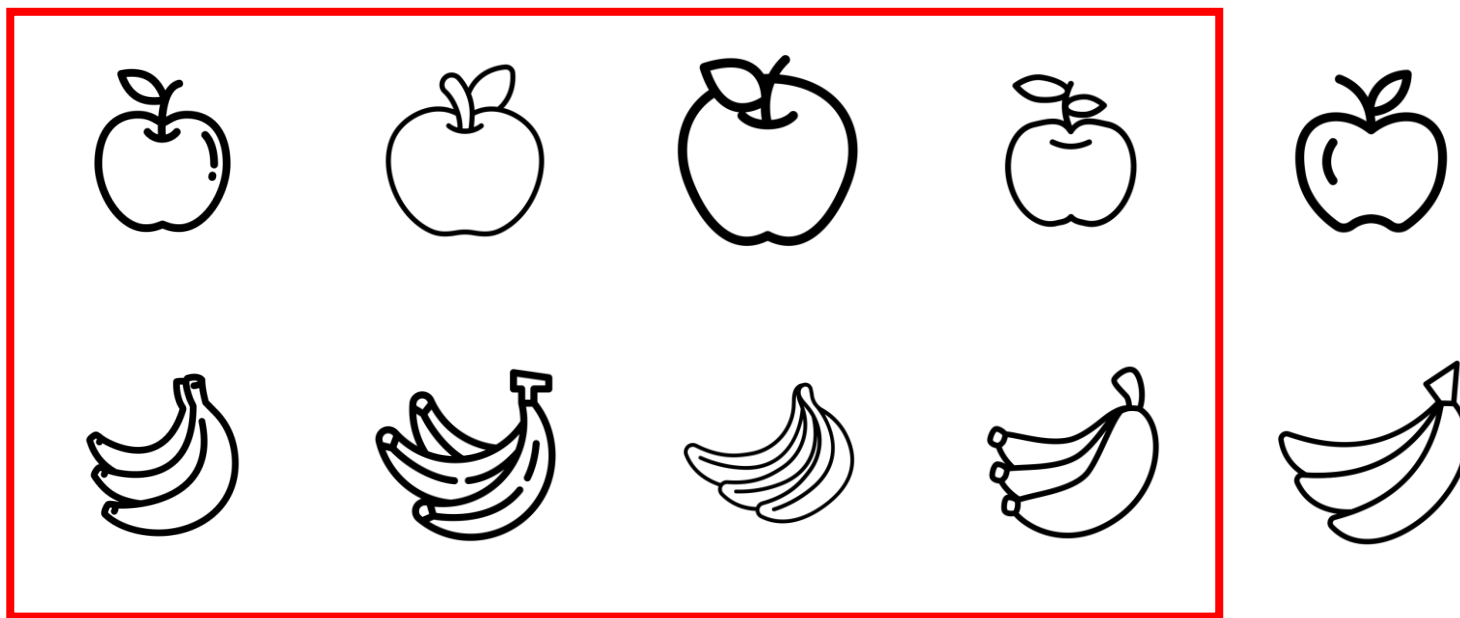
반복적 연습!



클리어 및 시간단축!

2-1. Train Set(훈련 세트)

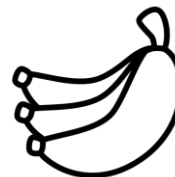
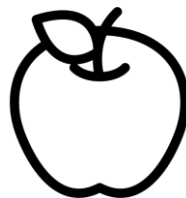
- 전체 데이터의 80% 정도를 train set으로 사용



train set

2-2. Test Set(테스트 세트)

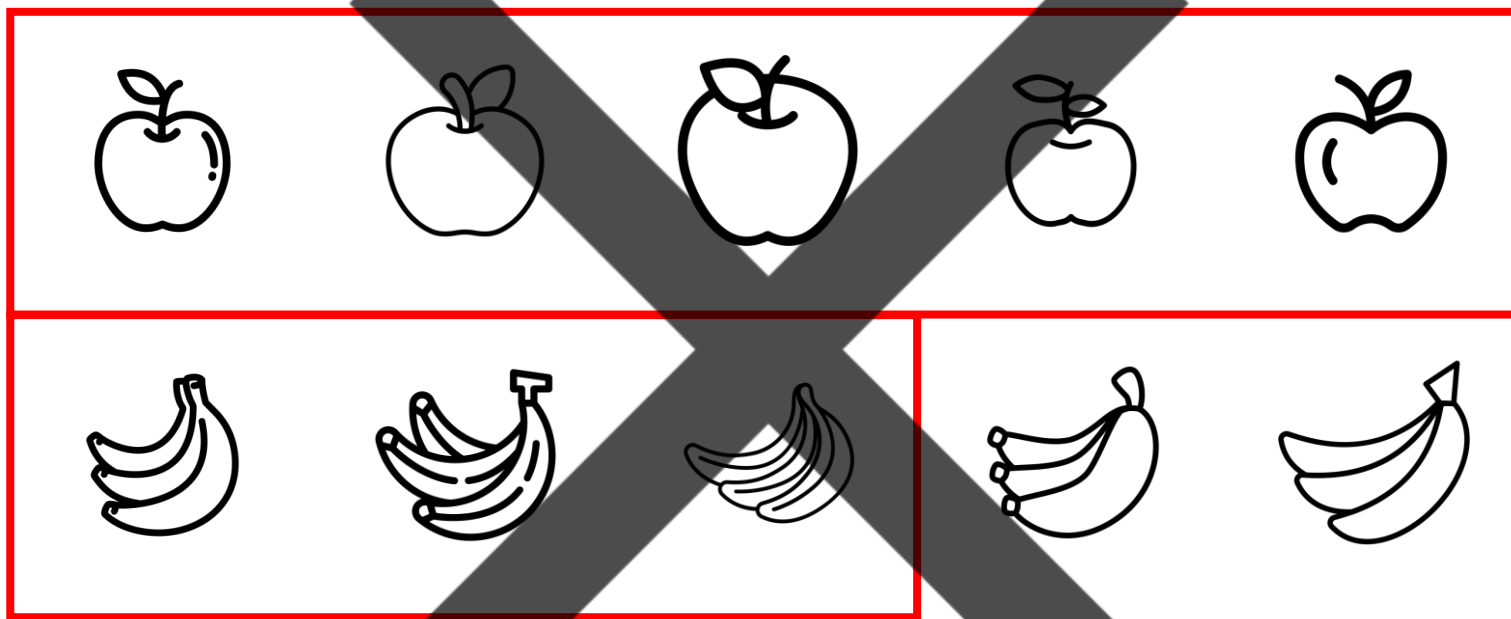
- 전체 데이터의 20% 정도를 test set으로 사용
 - 전체 데이터가 매우 클 때는 1%만 test set으로 사용해도 충분할 수 있음



test set

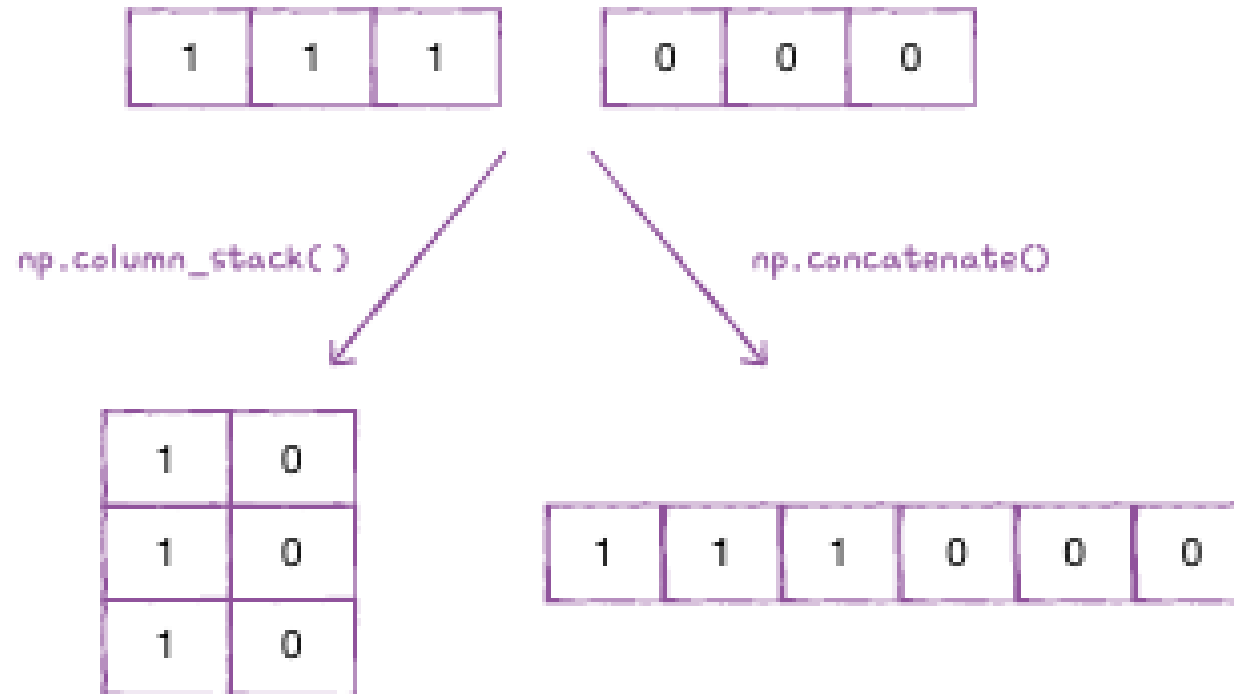
2-3. Sampling bias(샘플링 편향)

- train set과 test set을 나눌 때, **샘플이** 한쪽으로 치우치지 않고, **골고루 섞이도록** 주의



train set

Tip! column_stack / concatenate



3. Data Preprocessing(데이터 전처리)

10 vs 15 => 15가 더 크다!

하지만 10의 단위는 inch고, 15의 단위는 cm라면?
=> 두 특성은 scale이 다름

알고리즘이 거리 기반일 경우, 특성값을 일정한 기준으로 맞춰야 함 => 데이터 전처리
(트리 기반 알고리즘은 스케일이 달라도 잘 동작함, 나중에 배움)

가장 널리 사용하는 전처리 방법 : standard score(표준점수) = **Z score**
각 특성값이 평균에서 표준편차의 몇 배만큼 떨어져 있는지 나타냄

$$Z = \frac{x - \mu}{\sigma}, \mu : \text{mean(평균)}, \sigma : \text{std(표준편차)}$$

중요! 다른 샘플 데이터들을 train set의 mean과 std를 이용해서 변환해야함

Tip! scikit-learn 모델 사용법 간단 정리

```
# k-NN 알고리즘 사용 (기본 k값은 5)  
from sklearn.neighbors import KNeighborsClassifier
```

```
# 모델 생성
```

```
kn = KNeighborsClassifier()
```

```
# 모델 training
```

```
kn.fit(train_input, train_target)
```

```
# 모델 평가 (0 ~ 1사이의 값을 반환: 정확도(accuracy)) (ex- 1: 모든 데이터를 정확히 맞힘, 0.5 : 절반만 맞힘)
```

```
kn.score(test_input, test_target)
```

```
# 정답 예측
```

```
kn.predict(test_input)
```